CONNEXIONS™

The Interoperability Report

December 1990 Volume 4, No. 12

ConneXions — The Interoperability Report tracks current and emerging standards and technologies within the computer and communications industry.

In this issue:

ConneXions is published monthly by Interop, Inc., 480 San Antonio Road, Suite 100, Mountain View, CA 94040, USA. 415-941-3399. Fax: 415-949-1779.

ISSN 0894–5926

From the Editor

Well, it took about 6 months, but my plea for an X.25 article in our series *Components of OSI* has been answered. In this issue, Derek Vair of the Software Group in Canada presents an overview of X.25, which in fact covers 3 layers of the OSI stack: Network, Data Link and the Physical layer.

Fragmentation, the process by which a packet is split into two or more pieces in order for it to traverse a particular physical network, has been discussed in previous issues of *ConneXions.* (In particular, see *ConneXions,* Volume 2, No. 4, April 1988). Fragmentation is generally considered to be an "evil thing" as it can lead to severe performance degradation in internets. With this in mind, a working group of the Internet Engineering Task Force (IETF) set out to design a method by which the *Maximum Transmission Unit* (MTU) of a particular network path could be reported back to the sending host. If the MTU of a given path is known, the packet size can be adjusted by the sender to avoid any fragmentation. The mechanism developed by the working group is known as *Path MTU Discovery,* and was recently published as a Proposed Standard RFC. Jeff Mogul gives an overview of Path MTU Discovery on page 12.

The book *UNIX Network Programming* was reviewed in our April 1990 issue, and did not receive a very favorable review. More recently, I received another review of the same book, this time from a student of computer science who likens possession of *UNIX Network Programming* to having a "desktop guru." Therefore, I decided to publish this second review under the heading "Alternative Book Review," following the tradition started in our September issue.

Also in this issue, you will find a letter to the Editor in response to two articles which appeared in our September issue: "TCP and TP4: Moving Forward" and the Alternative Book Review of *The Open Book.* Mike Padlispsky, a real "Network Old Boy," takes Dave Piscitello, Lyman Chapin and Bryan Wood to task.

Since this is the last issue of the year, the new 1990 index sheet is included with this mailing. For your convenience, we have also enclosed index sheets for 1987, 1988 and 1989 as well as a subscription renewal and back issue order form. Let me remind you that subscription rates will be going up in January so you should renew your subscription now (even if it isn't due to expire soon) in order to take advantage of the old rates. We hope you have enjoyed this volume. Your comments and suggestions are always welcome.

We wish you a peaceful holiday, *ConneXions* will return with Volume 5 in a few weeks.

—Ole

Components of OSI: X.25—the Network, Data Link, and Physical Layers of the OSI Reference Model

by Derek Vair, The Software Group Limited

Overview

This article discusses the X.25 protocol and its applications. X.25 is the designation of a specification for a communications protocol adopted as a standard in the mid-1970s by the CCITT (*Comité Consultatif International de Télégraphique et Téléphonique*) for interfacing user equipment to a public packet-switching network. Subsequently, the ISO (*International Organization for Standardization*) adopted the protocol to provide the connection-mode network service of the OSI (Open Systems Interconnection) protocol suite.

The X.25 protocol both lacks some of the capability defined by the CCITT for a network service, and provides features beyond those required of the ISO network service.

X.25 origin

Every four years, the CCITT meets to review standards-creation activities. It issues each standard which it adopts as a *Recommendation* with an alphabetic/numeric designation. The X-series of Recommendations all have to do with connection to public data networks. For example, Recommendation X.25 describes the rules to be followed to connect a computer system to a public packet-switched data network. Rather than take the formal approach of discussing "the protocol defined by Recommendation X.25," the rest of this article simply refers to "the X.25 protocol."

As Tanenbaum [1] points out, the CCITT is not an open committee. It restricts voting memberships to government agencies, one per country. In most countries in the world, the voting government agency is the *Post, Telephone and Telegraph* (PTT) organization, which often holds a monopoly for providing telecommunications services in that country.

X.25 is a product of compromise, as it operates by consensus, rather than by majority voting. Every CCITT Recommendation contains the phrases "The CCITT considering,...unanimously declares the view..." The protocol reflects the fact that it was designed by a committee: it contains features and parameters of dubious utility which remain unimplemented or unused in practice.

Packet Retransmission, for example, an error-correcting feature introduced in 1980, remains unused by all the North American and most European packet-switched data networks.

Relationship to the OSI Reference Model

X.25 predates the OSI protocol suite by some time. It is a layered protocol, and conforms to the OSI reference model's layers I, II, and III. The X.25 PLP (*Packet Layer Protocol*—layer III) diverges from strictly providing OSI network services, as (among other things), it has no switching or routing functionality, and has aspects of end-to-end service that are not normally provided by a network layer.

The CCITT published X.25 as a means to control the interface between a customer's equipment (*Data Terminal Equipment,* or DTE) and a network administrators's switching node (*Data Communications Equipment,* or DCE). The protocol is also used for direct peer-to-peer communications, with no intervening packet-switched data network. When used in this manner, configuration control becomes critical: two identically-configured computers (i.e., two DTEs) cannot exchange information, as HDLC has specific protocol roles for each of the DTE and DCE.

Physical Layer (I)

The OSI physical layer is concerned with how to move bits down a wire. The details of electrical signalling, connector shape and size, and number of conductors are all part of this section of the specification. Recommendation X.25 borrows from existing standards, allowing a wide variety of electrical connections as Layer I:

X.21: is an electrical interface to a circuit-switched public data network which combines electrical signalling with a control language that allows devices to address the network directly.

X.21 bis: defines how the signals of a conventional (i.e., V-series) modem are used to establish a connection.

V-series modems: The CCITT publishes modem interface specifications which correspond closely to those published by the Electrical Industries Association. For example, RS-232-C and V.24 are parallel specifications.

X.31: defines the use of an Integrated Services Digital Network (ISDN) link for connection to the public packet-switched data network.

In practice, the link to an X.25 network today is a full-duplex leased line, using modems that conform to either RS-232-C (for low speed applications) or V.11/V.35 (for high speed ranges).

Recently, dialup access to public packet-switched data networks has become available. Using autodial modems, a DTE can establish a link to the network when required. Recommendation X.32 defines the service options for dialup X.25 access.

Data Link Layer (II)

An X.25 data link provides a transparent, error-free path for packets. It uses a version of the ISO *High Level Data Link Control* (HDLC) protocol for link setup, error recovery, and flow control on each physical link. To distinguish it from the original HDLC version used for X.25 (*Link Access Procedure,* or LAP), the HDLC protocol currently in use is referred to as LAPB, for Link Access Procedure, B.

X.25 can use more than one physical link to carry traffic. The protocol used to maintain proper sequencing with multiple physical links is called MLP (for *Multi-Link Protocol*). In the context of the reference model, it is a "layer 2.5," appearing between the X.25/PLP and LAPB protocols. Use of MLP protocol on a link is optional; there is very little, if any, commercial demand for this capability, and as a result, MLP testing isn't a part of the certification process to which X.25 implementations are subject prior to connection to many public packet-switched networks.

Packet Layer (III)

Unlike Layers I and II of X.25, the X.25 PLP was defined by the CCITT specifically for use in connecting to public packet-switched networks. The protocol allows management of up to 4095 separate logical connections multiplexed across the data link. Each of the slots available for a connection is called a *logical channel.* When logical channels are assigned for use in carrying data, they are usually referred to as *virtual circuits,* from the idea that each logical connection is a kind of imaginary circuit between two end points.

On any given link, the available logical channels can be defined to consist of permanent connections to other DTEs (*Permanent Virtual Circuits,* or PVCs), or as a common pool of temporary connections (*Switched Virtual Circuits,* or SVCs). The configuration of virtual circuits (how many, and for what use) is one of the parameters that a user organization and network vendor must agree to before a link becomes active.

continued on next page

Components of OSI: X.25 *(continued)*

Use of switched virtual circuits

The user's view of the telephone network is a good analogy to the use of an X.25 logical channel for transferring information. Consider the network as a cloud with a link from the cloud to each user. In the telephone network, the link is the pair of wires from the telephone to the nearest telephone company switching office, and the address is the phone number. In a packet-switching data network, the link is a leased line from the user's computer to the network vendor's switching node, and the address is a vendor-assigned string of digits associated with the link.

Using a logical channel to make a connection with an SVC is like making a telephone call. To start the call, a DTE sends a *Call Request* packet which contains the address of the remote DTE (goes off-hook and dials a number), and waits for a response (listens for ring). When the remote indicates it can communicate by responding with a *Call Accept* packet, the network sends the caller a *Call Connected* packet, (the other party picks up the telephone). At this point, the *Call Setup* phase of a Switched Virtual Circuit call is over, and *Data Transfer* begins.

Two DTEs use a virtual circuit to exchange data packets during the Data Transfer phase of the call, and one indicates the need to disconnect by sending a *Clear Request* packet (hanging up). The DCE confirms that the connection has been cleared and is now available for another call by transmitting a *Clear Confirmation* packet.

Call setup packets contain information beyond the addressing information required to reach another party connected to the network: *Facilities* fields and *Call User Data.*

Facilities

Facilities are a way of selecting options to be used while an SVC is connected. They can affect billing (both third party billing and collect calling are available on X.25 networks), class of service (effective throughput, transit delay, flow control parameters), security (closed user groups), and call detail recording (traffic data), among others.

The facilities field of call setup packets is the least stable element of the protocol. Each version of the specification contains new facilities to be implemented by both networking equipment vendors, and for network service vendors. New facilities needed to support the OSI connection-oriented network service have been added to both the 1984 and 1988 versions of the X.25 specification.

Call User Data

Call User Data is simply a field of bytes which the network passes unchanged from end to end. It contains coding to indicate what kind of upper layer protocol (X.3/28/29, RFC 877, QLLC, ISO Transport) to use, and an application-dependent area which can contain any information. For example, the Call User Data field can be used for port or application selection, or to carry a password for security-related applications.

The field is limited in size to either 16 or 128 bytes, depending on whether or not the network supports the datagram-like capability called *Fast Select.* Use of the Fast Select facility allows a calling DTE to present up to 128 bytes of information to be transferred to the remote system, which may respond with up to 128 bytes of *Clearing User Data.*

A Call Request using the Fast Select facility isn't a true datagram in the IP sense. There are serious protocol consequences if a Call Request packet gets lost. Logical channels can't be reused until there is an acknowledgement of an outstanding Call Request (a *Clear Indication* or *Call Connected* packet), or the calling DTE performs some kind of error recovery to regain access to the logical channel. There is a limit to the number of SVC calls that a DTE is allowed to have outstanding at any point in time. It can only send up to the maximum number of logical channels assigned to the link. This is really only of theoretical consequence considering that the number of logical channels available is 4096. However, since network vendors charge for making logical channels available, the reality is that logical channels are a contended-for resource, increasing the likelihood of running out of channels.

Permanent virtual circuits

Extending the telephone network analogy further, a PVC is similar to a "hot line" telephone connection with two hands-free telephone sets at each end. Speaking in one room always results in your words arriving at the other location. There is no call setup or clearing required for a PVC—it is always in the Data Transfer phase. PVCs don't require address information because the mapping between a specific logical channel number and a logical channel at the remote DTE is part of the network vendor's link configuration information, defined when the data link is installed.

Network interconnection

As for the worldwide telephone network, there is a scheme (X.121) for uniquely identifying each access line on a network. The CCITT assigns a *Data Network Identification Code* (DNIC) to each public data network. A DNIC has three digits which identify the country, and a single digit to identify the network. Although most countries have a single country code assigned, the CCITT has assigned seven country codes to the United States, allowing up to seventy public packet-switched networks with U.S. identification codes.

Network vendors handle the mechanics of interconnection, selecting which networks to connect to and supplying and operating the gateways between them. The protocol used for interconnecting public packet-switched networks is defined by a separate Recommendation, X.75, which is essentially X.25 rewritten with the term *Signalling Terminal Interface* (STE) replacing the familiar X.25 terms DTE and DCE, and reduced facility availability.

Like interconnected telephone networks, not all X.25 services are available across multiple public data networks. Collect calling isn't available from European networks to DTEs in North America, for example. The service available when data transits multiple networks is the lowest common denominator of services available on each connecting network.

There are public networks which cover more than one country, and their international charges tend to be lower than for traditional public data network international traffic. However, these networks are usually isolated—they have no official interconnection points to public data networks in the countries in which they operate.

Protocol evolution

Although the basic features defined in X.25 have changed very little (use of virtual circuits, sliding windows, and flow control in all protocol layers) there have been changes to the protocol every four years since the original specification was published. The 1976 document occupied less than 40 pages; the most recent version, published in 1988, is 159 pages long.

continued on next page

Components of OSI: X.25 *(continued)*

The following list shows the extensions made to the protocol with the various releases of the X.25 Recommendation:

1980:

- A Datagram-based mode of operation was introduced, where each datagram packet contains the source and destination address of each packet.
- New per-call facilities introduced:
  - Flow control negotiation
  - Fast Select

1984:

- Datagram packets were removed
- Multi-Link protocol was added
- Online facility registration
- Extensions to support the OSI Network layer defined:
  - Calling & called address extension facilities
  - Quality of service per-call facilities are added to support negotiation for minimum throughput class and end-to-end transit delay.
  - Expedited data negotiation

1988:

- Changes to per-call facilities:
  - Call deflection notification/indication
  - Extension to RPOA selection for route selection
  - Transit delay selection/indication
  - Eliminate "distance" parameter in charging information
- Additional quality of service extensions for OSI network layer support:
  - Priority
  - Protection

Application: Asynchronous PAD

The most common use for X.25-based public data networks was (and probably still is) connecting asynchronous terminals to host computers. When packet-switched networks first became available, network vendors had a serious problem: selling the service to users. Although a great deal of money had been spent on providing an alternative to conventional communications networks with a traffic based, rather than connection time based cost structure, there weren't any terminals which could use the network's native connection mechanism, X.25.

The solution to this problem lay in the development of a specialized protocol converter which could change the characters from the terminal into a stream of X.25 packets, and take X.25 packets and convert them into character streams.

X.3, X.28 and X.29 are the three CCITT Recommendations which specify how to convert a stream of asynchronous characters into packets, and vice-versa. The device defined by these recommendations is a *PAD* (*Packet Assembler/Disassembler*) for asynchronous terminals. An asynchronous PAD performs similar functions to the Telnet protocol and supporting software available on TCP/IP internets: connection establishment and control, and packetizing and depacketizing of keystrokes and terminal data. At the host end, a single X.25 link carries terminal traffic for a number of PAD-connected terminals.

The CCITT PAD depends on X.25 to provide a reliable transport service. It takes advantage of the end-to-end signalling mechanisms built in to X.25 to exchange protocol information with the host. Unlike Telnet, which has a mechanism whereby the devices at each end of a connection support a minimum option set (the *Network Virtual Terminal*) and can negotiate which enhancements to support during a connection, a CCITT PAD has no ability to negotiate the parameter set that it supports with a host.

PAD services are not limited to asynchronous terminals. There are PADs available for credit verification terminals, IBM BSC and SNA devices, Unisys terminal controllers, and many others. CCITT has only standardized the PAD for asynchronous terminals. Other PAD services which network vendors or connectivity equipment vendors may choose to provide are not generally available across networks.

Application: IP traffic

X.25 links can carry TCP/IP internet traffic, allowing both LAN interconnection and remote workstation connection to regional or central TCP/IP LANs. RFC 877 [4] describes how to use X.25 virtual circuits to carry IP traffic. Once a connection is established, IP packets are exchanged as complete data packet sequences. There is no additional protocol overhead beyond that of X.25—the first byte of the IP datagram is the first byte of the sequence of data packets.

Application: SNA

IBM's *Systems Network Architecture* (SNA) uses 3 link level technologies for communicating between hosts and terminal controllers or LU 6.2 peer entities: SDLC, QLLC, and the token-ring LAN. QLLC (*Qualified Logical Link Control*) combines with X.25 to provide the mechanism for transferring SNA data over public packet-switched data networks.

QLLC expects X.25 to provide reliable end-to-end transport of information. The existence of an X.25 virtual circuit is transparent to SNA upper layers, and (once the SNA link has been established via an XID protocol exchange), there is no overhead associated with use of the protocol—Request/Response Units make up complete data packet sequences in the same way as IP datagrams described above.

Application: Connection-oriented OSI Network Service

ISO 8878/CCITT X.223 defines how to use X.25 to provide a *Connection Oriented Network Service* (CONS). The table on the following page, extracted from the document, illustrates the mapping between the OSI primitives for *Network Connection* (NC) establishment and X.25 packet layer protocol elements.

The OSI service requires a subset of the set of the per-call facilities defined by the CCITT: the OSI-specific facilities, and Fast Select. Not all network vendors (of services or equipment) support all of these facilities in their implementations of the protocol, so data exchange between equipment or across more than one network may not be possible, as the usual response to an unrecognized facility code in a call request is to clear the call, making it impossible to establish a connection with the remote system.

continued on next page

Components of OSI: X.25 *(continued)*

CONS:X.25/PLP Mapping for the Network Connection Establishment Phase	
CONS	X.25/PLP
PRIMITIVES: N-CONNECT.request N-CONNECT.indication N-CONNECT.response N-CONNECT.confirm	PACKETS: CALL REQUEST INCOMING CALL CALL ACCEPTED CALL CONNECTED
PARAMETERS:	FIELDS: (INCLUDING FACILITIES)
Called address	Called DTE Address Field Called Address Extension Facility
Calling address	Calling Address Field Calling Address Extension Facility
Responding address	Called DTE Address Field Called Address Extension Facility
Receipt Confirmation Selection	General Format Identifier
Expedited Data Selection	Expedited Data Negotiation Facility
QoS-parameter Set	Throughput Class Negotiation Facility Minimum Throughput Class Negotiation Facility Transit Delay Selection And Indication Facility End-to-End Transit Delay Negotiation Facility
NS-User-Data	Call and Called User Data Field Fast Select Facility

Table 1: OSI CONS Primitive to X.25 packet type mapping for connection establishment.

For the other phases of a connection, the situation is similar. Only a subset of the available protocol capabilities are used, but the subset may not be part of the set of capabilities offered by a given service or equipment vendor. For example, the N_EXPEDITED DATA primitives use the X.25 Interrupt packet with an extended (up to 16 bytes long) data field. However, most public data networks don't provide this capability, so they don't have an expedited data service in their portfolio of services to provide to users.

Differences from the OSI Networking Service

CCITT X.25 offers a different set of capabilities (both more and less) from those required by a strict interpretation of the OSI definition for a networking service in X.213. There are three areas where X.25 network capabilities diverge from the OSI definition of a networking service:

- reliable end-to-end data data transport
- end-to-end significance of network layer protocol elements
- no packet routing or forwarding mechanisms

X.25's origin as a protocol for interfacing to communications facilities, rather than computer networking, account well for these differences. The first two items in the list above are direct results of the use of the protocol for carrying terminal traffic between a PAD and a host.

There is no guarantee that a given data packet will reach its destination, but packet loss in public data networks is very rare, for a very good reason: Packet network service vendors designed the networks to provide reliable data transport in order to sell their service. Potential customers would not switch from conventional communications techniques to another, less expensive, service, if they could expect a non-negligible error rate. It isn't an inherent feature of the protocol to provide reliable transport services, but rather the characteristics of the data networks which use X.25 interfaces.

As Padlipsky [2] points out, X.25 provides an end-to-end signalling capability as well as performing a network interface role. The header of an X.25 data packet allows the user to pass protocol control information end-to-end (by setting the *Q,* or data qualification, bit), and to cause acknowledgements to have end-to-end significance (by setting the D, or *Delivery Confirmation,* bit). This kind of end-to-end signalling is normally a characteristic of a transport layer protocol, rather than something from the network layer.

X.25 doesn't provide routing because the CCITT views packet routing as a function of the network switches, rather than something that user communications equipment should be involved in. Internetworking takes place at vendor-operated gateways, rather than in user computers. For companies with private packet-switching equipment, public network access is always a problem, because there is no simple mechanism provided by equipment vendors to access other networks, and public network vendors are reluctant to operate X.75 gateways with private companies (as opposed to other public network service vendors).

The use of X.25 to provide a connection-oriented networking service doesn't mean that the ISO has ignored routing. In the OSI stack, routing is to be performed in a networking sublayer operating above X.25, in much the same way that the Multi-Link Protocol operates above HDLC to provide support for multiple physical links. As Tsuchiya [3] points out, there are two adopted standards for *End System to Intermediate System* (ES–IS) routing information exchange specifications (one for each of the connection-mode and connectionless protocols), and work is in progress (but standards not yet adopted) on IS routing information exchange.

Recommendations and Reality

The services offered by public data networks are not synchronized with the CCITT Recommendations. Even though the technology may exist to support the 1988 version of CCITT X.25 in terms of the switch vendor's offerings, neither the service providers nor connectivity equipment vendors offer all the features of the protocol. For example, the following features of the protocol are currently not part of any public packet network vendor's offerings:

- Multi-Link Protocol
- REJ packets
- OSI Network Extensions

For users, this makes interoperability testing and careful specification review a critical part of any attempt to connect heterogeneous systems together using X.25 and/or public packet-switching networks. As network operators make the new features of the protocol available, this will become less important, as it is now for users of X.25 for applications other than OSI network communications.

continued on next page

Components of OSI: X.25 *(continued)*

Relevant Specifications

Comité Consultatif International de Télégraphique et Téléphonique, International Telecommunication Union, Geneva, Switzerland: 1976, 1980, 1984, 1988, Facscicle VIII.1: *Data communication over the telephone network:*

V.11: Electrical characteristics for balanced double-current interchange circuits for general use with integrated circuit equipment in the field of data communications.

V.24: List of definitions for interchange circuits between data terminal equipment (DTE) and data circuit-terminating equipment (DCE).

V.35: Data transmission at 48 kilobits per second using 60–198 kHz group band circuits.

Comité Consultatif International de Télégraphique et Téléphonique, International Telecommunication Union, Geneva, Switzerland: 1976, 1980, 1984, 1988, Facscicle VIII.2: *Data communication networks: services and facilities, interfaces:*

X.3: Packet assembly disassembly facility (PAD) in a public data network.

X.28: DTE/DCE interface for a start-stop mode data terminal equipment accessing the packet assembly/disassembly facility (PAD) in a public data network situated in the same country.

X.29: Procedures for the exchange of control information and user data between a packet assembly/disassembly (PAD) facility and a packet mode DTE or another PAD.

X.21: Interface between data terminal equipment (DTE) and data circuit-terminating equipment (DCE) for start-stop transmission services on public data networks.

X.25: Interface between data terminal equipment (DTE) and data circuit-terminating equipment (DCE) for terminals operating in the packet mode and connected to public data networks by dedicated circuit.

X.31: Support of packet mode terminal equipment by an ISDN.

X.32: Interface between data terminal equipment (DTE) and data circuit-terminating equipment (DCE) for terminals operating in the packet mode and accessing a packet switched public data network through a public switched telephone network or an integrated services digital network or a circuit switched public data network.

Comité Consultatif International de Télégraphique et Téléphonique, International Telecommunication Union, Geneva, Switzerland: 1976, 1980, 1984, 1988, Facscicle VIII.3: *Data communication networks:*

X.75: Packet-switched signalling system between public networks providing data transmission services.

X.121: International numbering plan for public data networks.

Comité Consultatif International de Télégraphique et Téléphonique, International Telecommunication Union, Geneva, Switzerland: 1976, 1980, 1984, 1988, Facscicle VIII.4: *Data communication networks: Open Systems Interconnection (OSI) model and notation, service definition:*

X.200: Reference model of open systems interconnection for CCITT applications.

X.210: Open System Interconnection (OSI) layer service definition conventions.

X.211: Physical service definition of open systems interconnection for CCITT applications.

X.212: Data link service definition for open systems interconnection for CCITT applications.

X.213: Network service definition for open systems interconnection for CCITT applications.

Comité Consultatif International de Télégraphique et Téléphonique, International Telecommunication Union, Geneva, Switzerland: 1976, 1980, 1984, 1988, Facscicle VIII.5: *Data communication networks: Open Systems Interconnection (OSI) protocol specifications, conformance testing:*

X.223: Use of X.25 to provide the OSI connection-mode network service for CCITT applications.

X.244: Procedure for the exchange of protocol identification during virtual call establishment on packet switched public data networks.

International Organization for Standardization, Geneva, Switzerland:

8208: Information processing systems—Data communications: X.25 Packet Level Protocol for Data Terminal Equipment.

8878: Information processing systems—Data communications: Use of X.25 to provide the OSI connection-mode network service.

ADDENDUM 1: Protection and priority.
ADDENDUM 2: Use of an X.25 PVC to provide the OSI CONS.

8881: Information processing systems—Data communications: Use of the X.25 packet level protocol in local area networks.

References

[1] A. S. Tanenbaum, *Computer Networks, Second Edition,* Prentice-Hall, ISBN 0-13-162959-X, 1988.

[2] M. A. Padlipsky, *The Elements of Networking Style,* ISBN 0-13-26811-001, Prentice-Hall, 1985.

[3] P. Tsuchiya, "Components of OSI: Routing (an Overview)," *ConneXions*, Volume 3, No. 8, August 1989.

[4] J. T. Korb, "Standard for the transmission of IP datagrams over public data networks," RFC 877.

DEREK R. VAIR received his B.A.Sc (Systems Design Engineering) from the University of Waterloo in 1975, and an M.B.A. from York University in 1984. Currently, he is a managing partner at The Software Group Limited, a UNIX communications technology company based in Woodbridge, Ontario. Prior to forming The Software Group, Derek was employed at Bell Canada, where he was responsible for design and implementation of one of the first X.25 PAD devices to be connected to the Datapac network.

[Ed.: Other articles in this series previously published include: ISDN, X.400, X.500, The Transport Layer, IS-IS Routing, ES-IS Routing, The Session Service, CLNP, The Presentation Layer, A taxonomy of the players, The Application Layer Structure, FTAM, The Security Architecture, and Group Communication. See *ConneXions* index pages for complete details].

Overview of Path MTU Discovery

by Jeffrey Mogul, Digital Equipment Corporation,
Western Research Laboratory

What is a Path MTU?

The Internet is a diverse collection of individual links connected by routers. For various reasons, link technologies limit the maximum IP packet size that they can carry: IP packets on Ethernets are limited to 1500 bytes, on FDDI networks to 4352 bytes, on slower Point-to-Point (PPP) links to 296 bytes, and so forth. The term *Maximum Transmission Unit* (MTU) refers to the maximum IP packet size that a link can carry (which may be slightly smaller than the data-link level maximum, since IP packets are usually preceded by data-link headers).

When an IP packet is forwarded by a router from one link to another, it is possible that the packet will be larger than the MTU of the outgoing link. In that case, the router must "fragment" the packet, into pieces none of which are larger than that MTU.

As an IP packet follows a path through the Internet, one or more of the links it traverses will have an MTU that is minimal for that path. We call that MTU the "Path MTU" (PMTU). Any packet larger than the Path MTU will be fragmented; other packets will follow the path without fragmentation.

Notice that we associate a PMTU with a directed path between a pair of hosts, not simply with the pair of hosts itself. That is because there may be many routes between a pair of hosts, each with its own PMTU, and two packets between the same pair of hosts may follow different paths. Most importantly, one cannot assume that packets from Host A to Host B follow the same path (in reverse) as packets from Host B to Host A. (The path also depends on the Type-of-Service, but mostly we can ignore that.)

Why do PMTU Discovery?

IP was intended to handle the diversity of link technologies, which is why fragmentation is a mandatory part of the IP architecture. One might suppose that fragmentation is sufficient to deal with the problem: a host picks a datagram size that suits its purposes, and blithely sends its packets, expecting that the routers will fragment as necessary, and the destination host will reassemble the pieces.

This is how it is supposed to work, but the network world is a dangerous place, and things go wrong. A few years ago, Chris Kent and I wrote a paper detailing some of the problems caused by fragmentation [1], which was summarized in *ConneXions* [2]. Briefly, fragmentation often leads to performance degradation, or even communication failure, through a mechanism we called "deterministic fragment loss" (DFL). The Internet is less susceptible to DFL than it once was, but in some cases it could still be a problem.

Fragmentation also can cause trouble for certain implementations of IP that are not able to reassemble datagrams. These implementations are broken, of course, but we should not put principle in the way of interoperability.

The 576 Rule

In our paper, we proposed a number of different approaches to resolve the problem, but the one that made the most sense at the time was the "576 Rule." This rule says that a host may use whatever datagram size it wants if the destination host is on the same subnetwork, but if there is a router on the path, then datagrams should be no more than 576 bytes.

The 576 Rule was crude and arbitrary, but it mostly worked, and was incorporated into many systems (including UNIX 4.3BSD and relatives). Its faults have become more severe with time, because more low-MTU links are being used (for SLIP and PPP connections) and because it is wasteful on the high-MTU links now used in the backbone networks. (If your packets are too small, then you are wasting all the bandwidth required to transmit the extra packet headers and the router resources necessary to switch the extra packets.)

For these reasons, it is now considered necessary for a host to make an informed choice of datagram size: the host must know the Path MTU. The existing IP architecture does not provide this information, so the IETF chartered a working group to devise a *Path MTU Discovery* protocol.

Goals

Our primary goal was that a new scheme be no worse than the old one. This meant that any pair of hosts that communicate using the 576 Rule should be able to communicate using the new scheme (and that any hosts that continued to use older schemes would not suffer, either.) Performance should be better in most cases, both for the hosts communicating and for the rest of the network. This meant that we could not make connection setup take a lot longer, we could not inject lots of extra packets into the network, and whatever we did could not destabilize the network under conditions of overload.

It was also important that our design support incremental adoption. We could not expect every host and router to be upgraded within the near future, and if the design needed most of the world to be upgraded before it had any effect, nobody would have an incentive to use it. In order to get people to adopt the design, we also had to keep it really simple. We also wanted a solution as soon as possible.

Dead ends

Prior to the chartering of the IETF working group, an attempt had been made to formalize one of the suggestions made in our paper. RFC 1063 [3] proposed a pair of IP header options; a host would put one of these options into the packets it sent, and the routers along the path would modify the option to reflect the Path MTU. The destination host would take the final value of the PMTU out of the received packet, and return this value to the sending host using the other option.

There were numerous problems with this approach. It required routers to handle a lot of packets carrying IP options, and many high-speed routers slow down whenever an option is present. It took up room in the IP packet, reducing the usable Path MTU. It required the receiving host to remember incoming options in order to return them to the sender, and the receiver had to decide which packet was to be used to carry the return option. And, it didn't work unless most routers were upgraded.

The first attempt of the IETF working group was not much more popular. It involved allocating one of the few unused bits in the IP packet header to mean "Report Fragments." A host participating in PMTU Discovery would set this bit on some of the packets it sent. If such a packet is ever fragmented, the receiver would send an ICMP message back to the sender; ICMP error messages contain enough information for the sender to know the size of the first fragment, which is a lower limit on the Path MTU.

continued on next page

Overview of Path MTU Discovery *(continued)*

The reason why we didn't want receivers to send fragmentation reports without "permission" from the sender is that this could fill the network with useless ICMP packets, either because the sender didn't understand their meaning (and so the fragments would continue ad infinitum), or because the sender had launched a burst of large packets, and so sending ICMP messages in response to each of the packets in the burst would be redundant.

This proposal quickly hybridized with RFC 1063, leading to a protocol in which senders occasionally send an IP header option asking routers to fill in the Path MTU value, and also asking receivers to report fragments. The hybrid proposal became distressingly complex.

All of these proposals shared one problem: they could cause fragments to be sent to those crippled IP implementations that did not do reassembly, in cases where currently (because of the 576 Rule) that would not happen.

The "Don't Fragment" approach

At this point, we discovered a solution that avoids most of the problems with our earlier efforts, yet is so simple that it barely deserves to be called a protocol.

One of the bits in the IP header is called "Don't Fragment" (DF); if a router would have to fragment a packet in order to forward it, but the DF bit is set, the router must drop the packet and notify the sender using an ICMP "Destination Unreachable/Fragmentation Needed and DF Set" message. (I prefer to call this a "Datagram Too Big" message.) Although this is part of the original IP specification, and all known routers implement it, we know of nobody who uses it.

A host using our approach to Path MTU Discovery simply sets the DF bit on every IP packet. With this one stroke, we guarantee that these packets will never be fragmented. Of course, in order to make progress from this point, the sending host must learn that its packets are being dropped, and decide how much to reduce the size of the datagrams it is sending. The Datagram Too Big message fulfills the first requirement; we propose to make it fulfill the second, too, by changing the specification of a previously "Reserved" field in this message. The new field would indicate the largest size packet that could have been forwarded to the destination in question (we call this the *Next-Hop MTU*).

Path MTU Discovery follows this simple procedure: a host starts by sending datagrams as large as will fit on the first-hop link, and if it receives a Datagram Too Big message, it ratchets down the datagram size and continues. If the path involves several decreases in MTU, it may take a few round trips before the correct size is chosen, but in practice "a few" should be no more than two or three. If the routing topology changes in such a way as to lower the Path MTU, the sending host (which is still setting DF on every packet) will find out right away.

It is possible that a path change may increase the Path MTU. We would like to discover this, so once every few minutes the host will increase its estimate of the Path MTU. Normally, the Path MTU will not have changed, and a few packets will be wasted as the old value is rediscovered, but if the Path MTU has increased, the larger packets will take advantage of it. It is important that a host not react too rapidly to an increase in Path MTU: if the path is oscillating between routes with different Path MTUs, we want the host to send packets that will make it through even the worst path.

What to do when a router doesn't cooperate

The design sketched above depends upon routers being upgraded to include the Next-Hop MTU value in Datagram Too Big messages. Current routers do not; they set this field to zero. It would appear that until all routers (or at least those that connect links of differing MTU) are upgraded, our scheme is useless. This is not so. Consider a host that receives an old-style Datagram Too Big message (distinguishable because the Next-Hop MTU field is zero, an impossible value). It knows that the packet in question was too big, so it can choose a lower Path MTU estimate, and try again.

We considered a few methods for choosing a new estimate. Suggestions included binary search and geometric progressions, but both of these suffer from slow convergence and inaccurate results.

It turns out that there are only a dozen or so MTUs actually in use in the Internet; this allows us to restrict the search to guesses that have a good chance of being right. We made a table of known MTUs, grouped the similar ones together, and chose a set of "plateaus" such that each plateau is at or slightly smaller than a known MTU. A host in this situation, then, simply chooses the next lower plateau for its new Path MTU estimate. If the estimate is right, then it is exactly right (or nearly so). If it is wrong, then the process repeats, but usually only a few times.

Status

Our proposal is not yet a standard. A draft was recently published as RFC 1191 [4].

In preparation for writing the draft, I did an implementation for 4.3BSD; this demonstrated that the mechanism works. Some of the problems I encountered in this implementation led to changes in the proposal, so my code isn't directly useful. Van Jacobson has also done an implementation, which should be available in 4.4BSD.

The draft proposal document includes quite a bit more detail than I could include in this brief overview; there are a number of subtleties in the design and implementation. Certainly, one should not attempt to implement Path MTU Discovery before reading the document.

Acknowledgements

Without the efforts of IETF *MTU Discovery Working Group,* including at times some sharp criticisms of earlier proposals, the mechanism described here would not have been developed (except that in its basic form it was originally suggested in 1987 by Geof Cooper, something that I remembered only after we had reinvented it). Steve Deering, in particular, made great sacrifices in order to work on MTU Discovery.

References

[1] Christopher A. Kent and Jeffrey C. Mogul, "Fragmentation Considered Harmful," Report 87/3, Digital Equipment Corporation Western Research Laboratory, November, 1987. Expanded version of paper presented at SIGCOMM '87.

[2] Jeffrey C. Mogul, "Fragmentation: Pros and Cons," *ConneXions—The Interoperability Report,* Volume 2, No. 4, April, 1988.

[3] J. Mogul, C. Kent, C. Partridge, K. McCloghrie, "IP MTU Discovery Options," RFC 1063.

[4] J. Mogul & S. Deering, "Path MTU Discovery," RFC 1191.

JEFFREY C. MOGUL received an S.B. from M.I.T. in 1979, an M.S. from Stanford in 1980, and his PhD from the Stanford Computer Science Department in 1986. While at Stanford he produced the distribution of the Stanford implementation of PUP protocol software for UNIX. He is author or co-author of a number of RFCs, including those specifying Internet Standards for subnets, broadcasting, and RARP. Since 1986, he has been a researcher at DEC Western Research Lab, working on network and operating systems issues for high-performance computer systems.

Alternative Book Review

UNIX Network Programming, by W. Richard Stevens, Prentice-Hall, 1990, ISBN 0-13-949876-1. Previously reviewed in *ConneXions* Volume 4, No. 4, April 1990.

This book fills a gap in networking literature. Due to the complexity and diversity of networking, most books tend to specialize. This book, however, provides an essential source that covers networks from the grand scheme of things to the details of implementing networking software for both Berkeley and System V versions of UNIX. For the novice network programmer, this book will provide a learning text which satisfies curiosity from the very basic concepts of networking through to the exact details of implementing networking software. For the experienced network programmer, this book is a time-saving reference filled with specific and practical example source code.

Organization

The book is written in a very readable style, and presupposes only a limited knowledge of **C** and a working knowledge of UNIX. It is comprised of over 700 pages with over 15,000 lines of well documented source code (about 234 pages), and is divided into four basic sections: IPC on a single host, internetworking, programmatic interfaces such as Berkeley Sockets and System V TLI, and example applications.

As seems to be the rule with books about UNIX, the first section serves as an overview of UNIX including such things as processes, shells, user-IDs, **C**, and network related system calls. The latter half of this section becomes more focused, covering methods of Inter-process Communication on a single system. This includes record/file locking, pipes, FIFOs and semaphores.

Internetworking

The second section of the book deals with how information is passed between machines. This information includes the hardware: LANs, gateways, WANs, bridges, routers, and repeaters; as well as over-views of different protocols: TCP/IP, XNS, OSI, and more. This section gives the programmer an idea of what goes on behind the system and library calls used to establishing connections and transferring data.

Programmatic interfaces

Section three deals specifically with the interfaces an application will use to communicate across a network. This covers Berkeley's *Sockets*, and System V's TLI (*Transport Layer Interface*). It contains a description of the calls and compilable source code that demonstrates how to use sockets or TLI.

Example applications

The final section is by far the most useful learning tool, and comprises most of the text (about 43 percent of the book). This section presents network utility routines for establishing connections, source code for *ping, rlogin, lpr, tftp* (trivial FTP) and subsections on security, remote command execution, remote procedure calls and more. The source code provided in the section is well documented and primarily written for Berkeley Socket's, but include discussions of System V's TLI. With the inclusion of Berkeley Sockets in System V.4, more of the source code in this section will be useful to people in the AT&T domain.

Good points

The book contains a wealth of knowledge about networking. In the past to find needed information, such as how to use sockets or pseudo-terminals, one would have to read through dated, very terse documentation that often provided incomplete example code.

This text, with its complete examples, ends the frustration caused by having documentation that raises more questions than it answers. Due to its great scope, it is able to cover some rarely known features, such as passing file descriptors between processes (and gives an example program for doing so). The example programs along with their explanations make this book well worth purchasing. The availability of the examples in this book via anonymous FTP from `uunet.uu.net` is also helpful. For experienced network programmers, this text provides a solid tool.

Bad points

The structure of the book seems a bit clumsy. For example, the demonstration of how to use pseudo-terminals is presented in the section covering rlogin, rather than in a section that covers IPC on a single host. Also, in order to be able to cover the largest amount of information, detail for the beginner is sometimes overlooked.

Recommended

I very highly recommend this book. As previously stated, this book helps to fill a gap in the literature. Its like having a desktop guru. This book contains so much information that, like the guru, you can keep coming back to get your questions answered. Perhaps from now on, I'll stop seeing a flood of people posting to `comp.unix.questions` asking how to use Sockets or pseudo-terminals. —*Eli Charne*

A Letter to the Editor

Dear Ole:

Us and Them

Although I have a certain amount of sympathy for your stated position with respect to cooperation between Us (The Internet Camp) and Them (The ISO Side), I believe you are far too kind to Them in overlooking the extent to which We have been the injured party. So with apologies for feeling constrained to continue the Us/Them shorthand and for the lack of customary levity (both dictated by your space constraints—at least I don't *think* I've lost my sense of humor), I'd like to respond to your September 1990 [Volume 4, No. 9] issue.

Consider the "Alternative Book Review." Just as I have heard aimed at myself even since before I coined the term "ISORMite," it's the same old story still:

Say you disagree
And you don't understand
Say "I disagree"
And you're self-indulgent

And here it's aimed at a book that's fundamentally on Their side! I suggest that They don't want cooperation, They want capitulation.

Reference Models

Now consider the TCP/TP4 article. At least it's not condescending, but that doesn't make it correct. It is *not* the case that the "architectures" are the same below Transport/Host-Host; the ISORM views the Network layer as end-to-end, whereas the ARM (ARPANET Reference Model) views it as an interface. Thus, the TP "family" takes cognizance of the properties of "the" network and offers non-interoperable members of the family depending on the perceived properties of "the" network, while TCP offers a reliable Host-Host/Transport mechanism and UDP offers a mechanism for situations where reliability is not required, *in the context of a "Catenet"* (in the original sense of the term).

continued on next page

A Letter to the Editor *(continued)*

Different contexts

Another way of putting it is that the ARM protocols were developed for use in a Catenet environment and ISORM protocols weren't. ("ISO IP" is an add-on, and still hasn't been successfully integrated by all reports.) In any event, for all their apparent similarities TCP and TP4 are intended to operate in such different contexts that it is impossible for me to call them functionally equivalent in a deep sense.

Couple the false premises with the flawed reasoning about "how it implements" (where the issue isn't that Their implementations aren't "seasoned" but rather how much more mechanism one is expected to buy off on in order to get the allegedly richer functionality—which almost all turns out to be optional anyway), and the lead article comes off as being a rationalization for capitulation (though I'm *not* accusing the authors of being quislings, merely of being too well intentioned) rather than a rationale for cooperation, in my humble-but-dogmatic opinion.

Revise the OSI Reference Model!

To risk using the dreaded first-person singular, I'd like to suggest that however politically infeasible it might be to effect, it really does seem that the best project to cooperate on would be a revision of Their "reference model" (RM).

This would serve two extremely important purposes: First, it would eliminate the real or feigned superstitious awe They affect, which would make Them far less distasteful to deal with. Second, it would require straightening out the misaligned protocols They've come up with in order to make them comply with the corrected RM, which should make them worth implementing. (One hopes.)

Note, by the way, that it has been the rigidity of Their "all n-entities must communicate with their peer n-entities via n-1 entities" hierarchical-ism that led Them astray all the long at Layer [sic] 5–7, which in turn led to the artificialities of all those epicyclic "ASEs" that at least according to your alternative reviewer even Marshall Rose hasn't been able to keep track of. So it isn't just to deal with the differences between TCP and TP4 that the sacred writ needs to be clarified by a suitable ecumenical effort.

Dogged cheers,

Mike Padlipsky

Mr. Padlipsky, who also coined the phrase "Old Network Boy," as usual withholds the identity of his current employer, lest anybody imagine even for an instant that his views are other than his own. As always, we welcome your comments and suggestions —Ed.

Upcoming Events

The USENIX *Symposium on Experiences with Distributed and Multiprocessor Systems* (SEDMS) will be held March 21–22, 1991, Atlanta, GA. The symposium is sponsored by The USENIX Association in association with the NSF/Purdue/Florida Software Engineering Research Center, and in cooperation with ACM SIGCOMM and SIGOPS, and the IEEE-CS Technical Committees on Distributed Processing, Operating Systems, and Software Engineering.

Background

In October of 1989, the *Software Engineering Research Center* (SERC) and USENIX cosponsored a very successful workshop devoted to Experiences with Distributed and Multiprocessor Systems. The ACM and IEEE-CS served as cooperating sponsors.

Format

Based on that experience, we are going to hold the event again. Because of the number and quality of submissions the first time, and the expected number of attendees, we have decided to hold the second instance of this event as a more structured symposium rather than as a workshop. As at the first workshop, we will have extra-long breaks between sessions and evening discussion events to promote a workshop-like atmosphere, but with attendance of over 200 a distinct possibility, we must plan for something a bit more formal than a workshop.

We have decided to hold the SEDMS on the East Coast again this year to facilitate the participation of European and South American researchers. We were gratified at the number of non-US researchers who submitted papers and attended the first workshop. We would like to do what we can to encourage a high level of international participation, and our survey of attendees indicated an East Coast location would help.

Publication

A special issue of the journal *Computing Systems* (Winter 1990) contains 5 articles developed from papers presented at the first workshop. We hope to do something similar with significant submissions from the 1991 SEDMS.

Goals

The goal of this symposium is to bring together individuals who have built, are building, or will soon build distributed and multiprocessor systems, especially operating systems. The symposium will feature full presentations and (perhaps) panels on aspects of building, testing, debugging, and using these systems. We will provide a forum for individuals to exchange information on their experiences, both good and bad, including experiences with coding aids, languages, distributed debugging tools, prototyping, reuse of existing software, performance analysis, and lessons learned from use of such systems. For further information, contact:

George Leach, General Chair
AT&T Paradyne
MS LG-129
PO Box 2826
Largo, FL 34649-2826
813-530-2376
reggie@pdn.paradynecom

Gene Spafford, Program Chair
Software Engineering Research Cntr
Dept. of Computer Sciences
Purdue University
West Lafayette, IN 47907-2004
317-494-7825
spaf@cs.purdue.edu

To have your name added to the mailing list for registration materials, send your name and surface mail address to either the general chair or program chair.

For more information about this or other USENIX conferences, contact the USENIX office: office@usenix.org, or 415-528-8649.

CONNEXIONS
480 San Antonio Road
Suite 100
Mountain View, CA 94040
415-941-3399
FAX: 415-949-1779

ADDRESS CORRECTION
REQUESTED

Bulk Rate
U.S. POSTAGE
P A I D
SAN JOSE, CA
PERMIT NO. 1